Transformación de Modelo Conceptual a Relacional

Universidad de Sevilla

Departamento de Lenguajes y Sistemas Informáticos

Objetivos


Asociaciones
Modelo
Clave
Tabla
Atributos
Muchos
Dirigido
Entidades
Modelos
Relaciones
Transformación
Contenido
Objetos
Relación
Nulos
Claves
Desarrollo
Clasificaciones
Clase
Discriminante

Contenido

Contenido

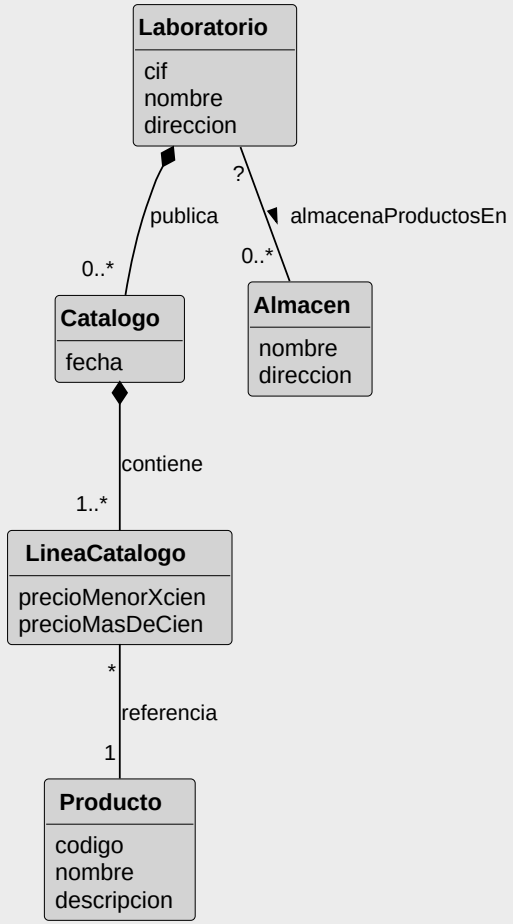
Desarrollo dirigido por modelos (MDD)

Enfoque de desarrollo en el que se van generando productos mediante transformación de modelos hasta llegar al código fuente.

En el desarrollo de sistemas de información con BBDD relacionales, se puede transformar el modelo conceptual para obtener un modelo relacional que luego puede transformarse para obtener código SQL.

Desarrollo dirigido por modelos (MDD)

Modelo conceptual



Modelo Relacional

```
Laboratorios = { laboratorioId, cif, ... }
  PK(laboratorioId)
  AK(cif)
Almacenes = { almacenId, nombre, dirección }
  PK(almacenId)
AlmacenesLaboratorios = { almacenLaboratorioId,
        almacenId,
        laboratorioId }
  PK(almacenLaboratorioId)
  FK(almacenId)/Almacenes
  FK(laboratorioId)/Laboratorios
  AK(almacenId, laboratorioId)
```

Modelo Tecnológico (SQL)

```
CREATE TABLE Laboratories (
  laboratory_id INT AUTO_INCREMENT NOT NULL,
  cif VARCHAR(20) NOT NULL,
  lab_name VARCHAR(120) NOT NULL,
  address VARCHAR(255) NOT NULL,
  PRIMARY KEY (laboratory_id),
  UNIQUE (cif)
);
```

Transformación de entidades

Regla general:

- Cada entidad se transforma en una relación con su mismo nombre (en plural).
- Cada atributo de la entidad se transforma en un atributo de la relación con su mismo nombre.

A tener en cuenta:

- Definir dominios para los atributos en función de la semántica de los atributos correspondientes.
- Añadir un sufijo Id como clave primaria (PK).

Transformación de entidades

Identificadores de objetos (ID)

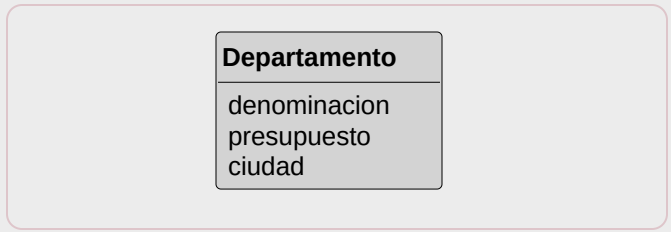
- Son atributos artificiales* que se añaden para poder identificar un objeto de otro sin depender de las claves semánticas. Denominados habitualmente **surrogates** (sustitutos) en inglés, suelen ser numéricos.
- Normalmente se delega la generación de sus valores al SGBD (autonuméricos, secuencias, etc.).
- Se suelen nombrar con el sufijo Id (usuarioId, pedidoId, alumnoId, ...)

Claves semánticas

- Son atributos o conjuntos de atributos que son claves en el dominio del problema.
- A veces, pueden usarse como Id porque realmente los son en el dominio del problema: NIF, NSS, código de producto, etc.

Contenido

Transformación de entidades




```
-- Intensión ------------------------------------------------------------------
Departamentos = { departamentoId, denominación, presupuesto, ciudad }
    PK(departamentoId)
    AK(denominación)

-- Extensión ------------------------------------------------------------------
Departamentos = {
    ('D1', 'Historia', 20000, 'Sevilla'),
    ('D2', 'Arte'    ,  5000, 'Sevilla'),
    ('D3', 'Dibujo'  ,  5500, 'Cádiz'),
    ...
}
```

Contenido

- Desarrollo dirigido por modelos
- Transformación de entidades
- **Transformación de asociaciones**
- Transformación de clasificaciones
- Ejemplos de transformación

Transformación de asociaciones

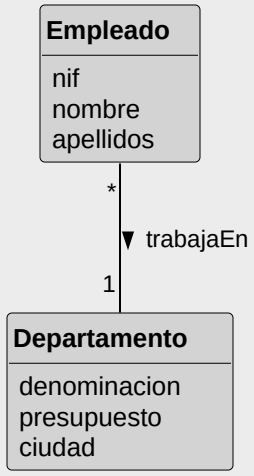
Regla general:

- Las asociaciones **1:N** se representan con una clave ajena en la relación de la entidad del rol N.
- Las asociaciones **1:1** se representan con una clave ajena en cualquiera de las relaciones.
- Las asociaciones **M:N** se representan con una relación auxiliar con claves ajenas a las dos relaciones.

A tener en cuenta:

- Si un rol está {**ordenado**}, hay que añadir un atributo (si no existe ya) que especifique el orden en la misma relación en la que se coloca la clave ajena.

Transformación de asociaciones 1:n



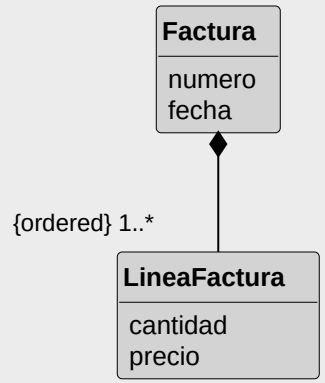

```
-- Intensión ------------------------------------------------------------------
Departamentos = { departamentoId,denominación, presupuesto, ciudad }
    PK(departamentoId)
    AK(denominación)

Empleados = {empleadoId, departamentoId, nombre, apellidos}
    PK(empleadoId)
    FK(departamentoId)/Departamentos

-- Extensión ------------------------------------------------------------------
Departamentos = {
    ('D1', 'Historia', 20000, 'Sevilla'),
    ('D2', 'Arte'    ,  5000, 'Sevilla'),
    ('D3', 'Dibujo'  ,  5500, 'Cádiz')
}

Empleados = {
    ('E1', 'D1', 'Luis' , 'Ortega'),
    ('E2', 'D1', 'Juan' , 'Pérez'),
    ('E3', 'D1', 'Sofía', 'López'),
    ('E4', 'D2', 'Lucas', 'Ruiz')
}
```

Transformación de asociaciones 1:n con orden



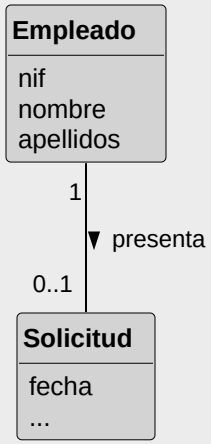

```
-- Intensión ------------------------------------------------------------------
Facturas = {facturaId, número, fecha}
    PK(facturaId)

LíneasFactura = {lineaFacturaId, facturaId, orden, cantidad, precio}
    PK(lineaFacturaId)
    FK(facturaId)/Facturas
    AK(facturaId, orden)

-- Extensión ------------------------------------------------------------------
Facturas = {
    ('F1', 410923, '2019-09-19'),
    ('F2', 410954, '2019-09-23')
}

LíneasFactura = {
    ('LF1' , 'F1', 1, 23, 78.23),
    ('LF2' , 'F1', 2, 51, 52.34),
    ('LF3' , 'F1', 3, 25, 63.15),
    ('LF12', 'F2', 1, 33, 44.12),
    ('LF13', 'F2', 2,  2, 55.12),
    ('LF14', 'F2', 3, 15, 34.22)
}
```

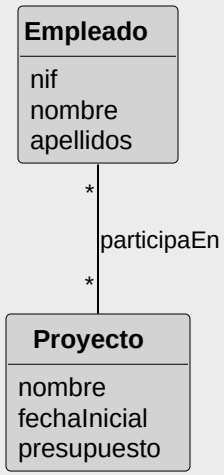
Transformación de asociaciones 1:1




```
-- Intensión ------------------------------------------------------------------
Empleados= { empleadoId, nif, nombre, apellidos}
  PK(empleadoId)
  AK(nif)
Solicitudes = {solicitudId, empleadoId, fecha}
  PK(solicitudId)
  FK(empleadoId) / Empleados
    AK(empleadoId)  -- Implementa cardinalidad 0..1

-- Extensión ------------------------------------------------------------------
Empleados = {
    ('E1', '25364987S', 'Luis' , 'Ortega'),
    ('E2', '25639874E', 'Juan' , 'Pérez'),
    ('E3', '89652314R', 'Sofía', 'López'),
    ('E4', '78541254G', 'Lucas', 'Ruiz')
}
Solicitudes = {
  ('S1', 'E1', '14-02-2026'),
  ('S2', 'E3', '15-02-2026')
}
```

Transformación de asociaciones m:n




```
-- Intensión ------------------------------------------------------------------
Empleados = {empleadoId, nif, nombre, apellidos}
    PK(empleadoId)
    AK(nif)
Proyectos = {proyectoId, nombre, fechaInicial, presupuesto}
    PK(proyectoId)
EmpleadosProyectos = {empleadoProyectoId, empleadoId, proyectoId}
    PK(empleadoProyectoId)
    FK(empleadoId) / Empleados
    FK(proyectoId) / Proyectos
    AK(empleadoId, proyectoId)
-- Extensión ------------------------------------------------------------------
Empleados = {
    ('E1', '25364987S', 'Luis' , 'Ortega'),
    ('E2', '25639874E', 'Juan' , 'Pérez'),
    ('E3', '89652314R', 'Sofía', 'López'),
    ('E4', '78541254G', 'Lucas', 'Ruiz'), ... }
Proyectos = {
    ('P1', 'Hércules', '2018-04-12', 234000),
    ('P2', 'Apollo',   '2019-01-27', 543000), ... }
EmpleadosProyectos = {
    ('EP1', 'E1', 'P1'),    ('EP2', 'E3', 'P1'),    ('EP3', 'E2', 'P1'),
    ('EP23', 'E2', 'P2'),   ('EP25', 'E4', 'P2'),   ... }
```

En la relación EmpleadosProyectos la pareja de claves ajenas también es clave alternativa

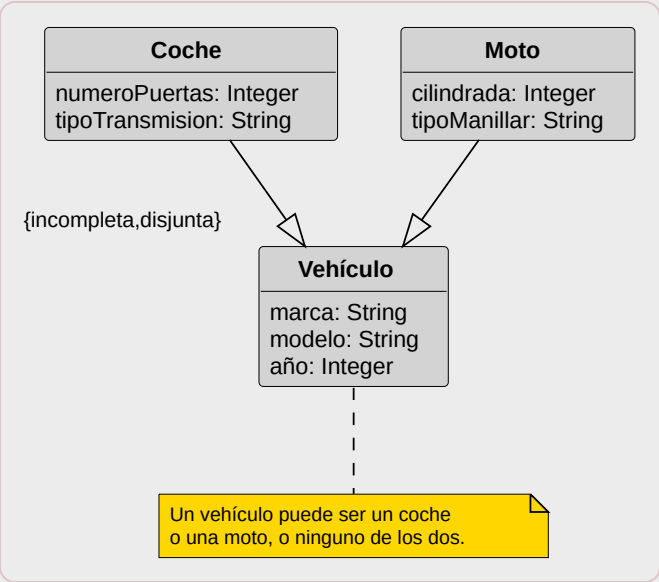
Contenido

Clasificaciones: Estrategias

Estrategias de transformación:

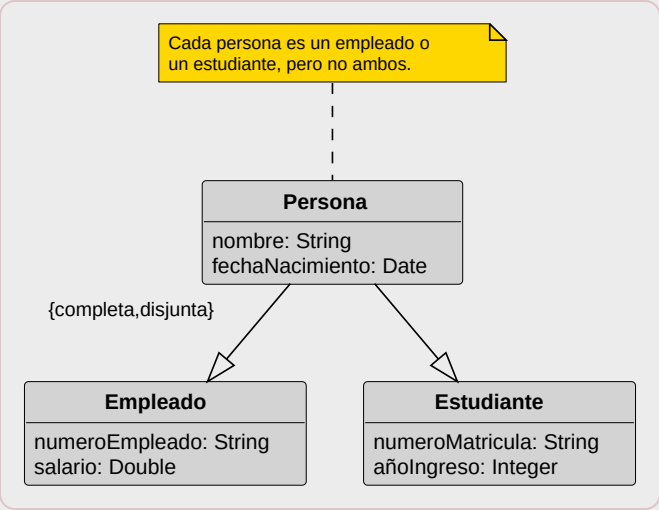
- Una relación para cada clase de la jerarquía. (Incompleta/completa)
- Una relación para cada subclase concreta. (Completa)
- Una única relación para toda la jerarquía. (Completa/Incompleta)
  - Con discriminante (Disjuntas)
  - Con booleano (Solapadas)

Clasificaciones: Una relación por clase



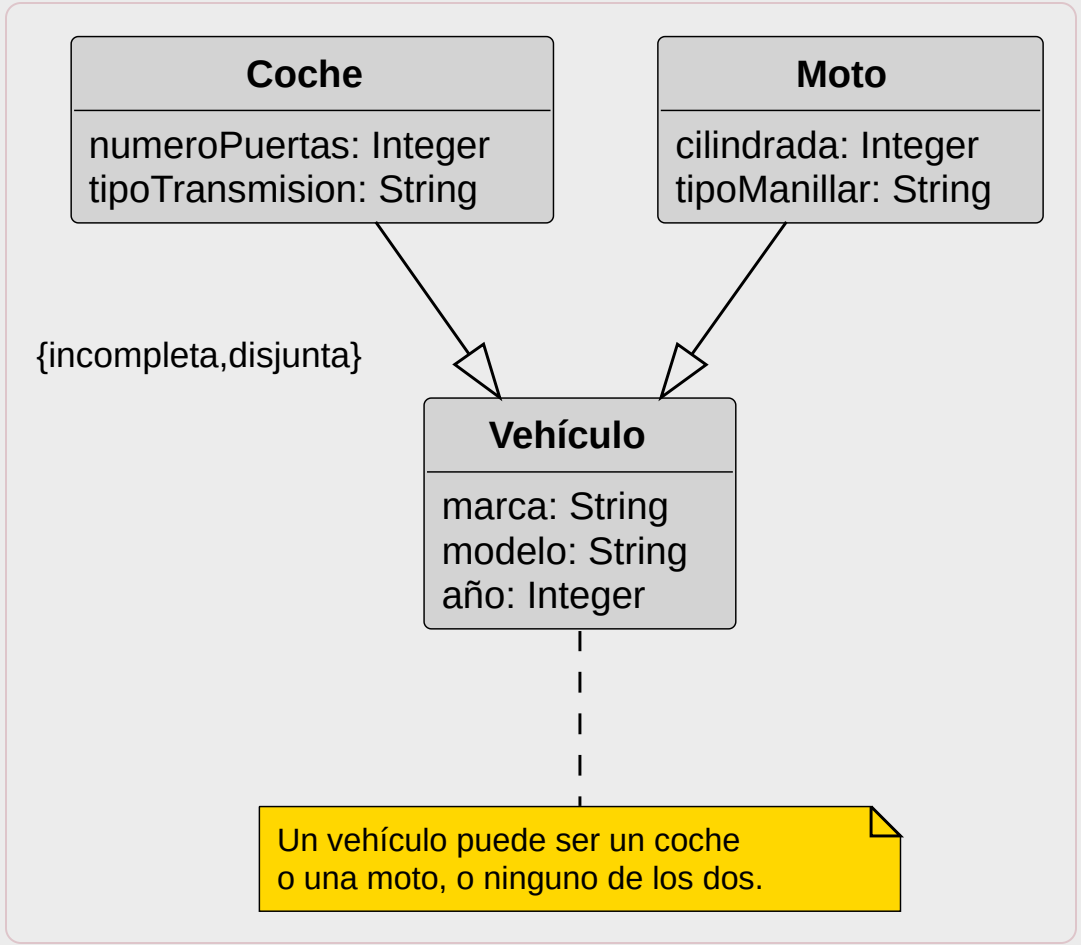

```
-- Intensión  ------------------------------------------------------------
Vehículos(vehículoId, marca, modelo, año)
  PK(vehículoId)
Coches(vehículoId, númeroPuertas, tipoTransmisión)
  PK(vehículoId)
  FK(vehículoId) / Vehículos
Motos(vehículoId, cilindrada, tipoManillar)
  PK(vehículoId)
  FK(vehículoId) / Vehículos
-- Extensión ------------------------------------------------------------
Vehículos = {
  ( v1, 'Ford'    , 'F-150'   , 2021), ( v2, 'Toyota'   , 'Corolla'  , 2022),
  ( v3, 'Honda'   , 'CBR'     , 2023), ( v4, 'BMW'      , '320i'     , 2023),
  ( v5, 'Yamaha'  , 'R1'      , 2022), ( v6, 'Chevrolet', 'Silverado', 2020),
  ( v7, 'Audi'    , 'A4'      , 2023), ( v8, 'Kawasaki' , 'Ninja'    , 2021),
  ( v9, 'Mercedes', 'Sprinter', 2022), (v10, 'Honda'    , 'Civic'    , 2023),
  (v11, 'Ducati'  , 'Panigale', 2023), (v12, 'Nissan'   , 'Titan'    , 2021)
}
Coches = {
  ( v2, 4, 'AUTO'),  ( v4, 4, 'Manual'),
  ( v7, 2, 'AUTO'),  (v10, 4, 'CVT')
}
Motos = {
  ( v3,  600, 'SPORT'),   ( v5, 1000, 'SPORT'),
  ( v8,  650, 'Cruiser'), (v11, 1200, 'SPORT')
}
```

Clasificaciones: Una relación por subclase concreta




```
-- Intensión ------------------------------------------------------------------
Empleados = { personaId, nombre, fNacimiento, numEmpleado, salario }
    PK(personaId)
    AK(numEmpleado)
Estudiantes = { personaId, nombre, fNacimiento, numMatricula, añoIngreso }
    PK(personaId)
    AK(numMatricula)
--Extensión -----------------------------------------------------------------
Empleados = {
    (p1, 'Juan Pérez'   , 1985-03-15, 'E001', 50000.0),
    (p3, 'Carlos Ruiz'  , 1978-11-08, 'E002', 65000.0),
    (p5, 'Roberto Silva', 1982-09-30, 'E003', 58000.0),
    (p7, 'David Torres' , 1975-06-25, 'E004', 72000.0)
}
Estudiantes = {
    (p2, 'Ana García'   , 2000-07-22, 'M2023001', 2023),
    (p4, 'María López'  , 1999-04-12, 'M2023002', 2023),
    (p6, 'Laura Martín' , 2001-01-18, 'M2024001', 2024),
    (p8, 'Elena Vázquez', 2000-12-03, 'M2024002', 2024)
}
-- LOS IDS DE LAS SUBCLASES DEBEN SER DISJUNTOS
```

Clasificaciones: Una relación con discriminante

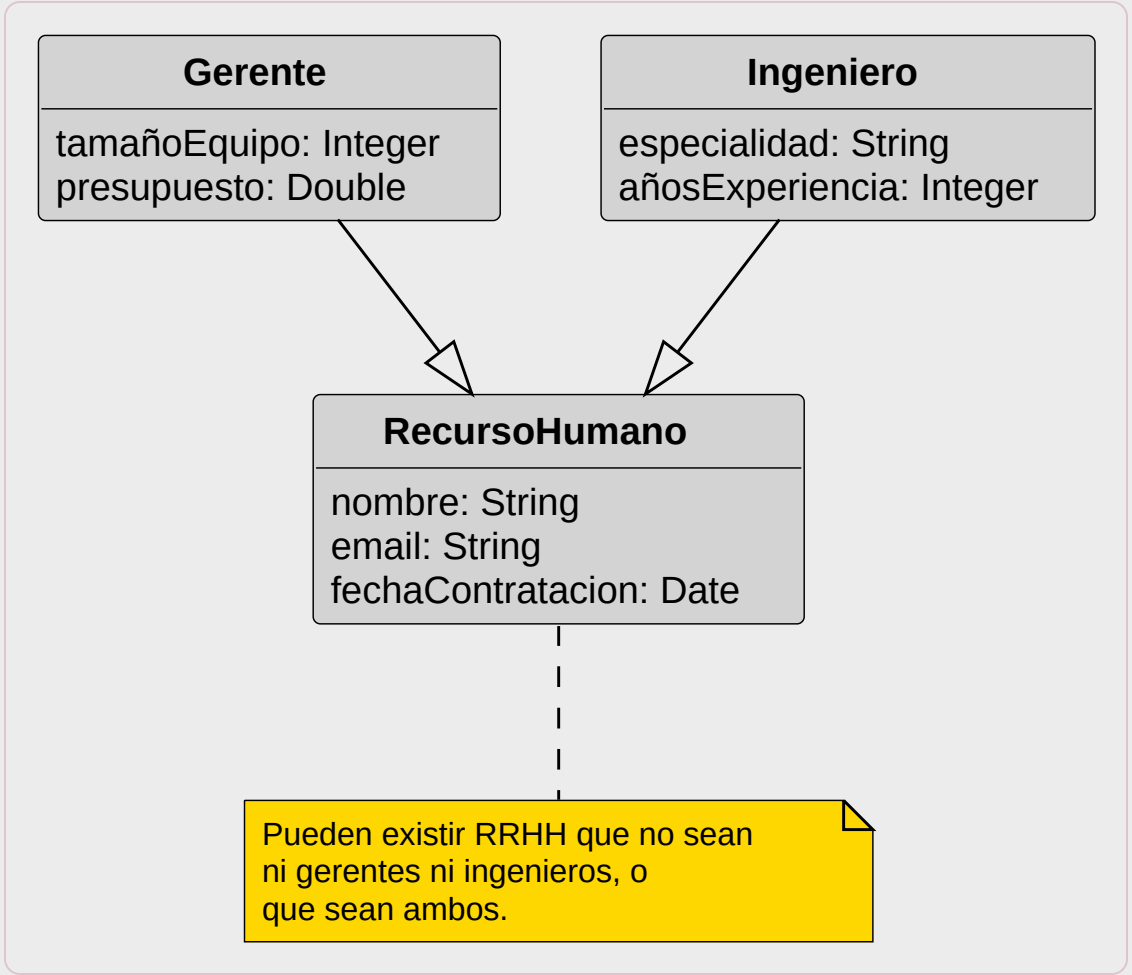


Clasificaciones: Una relación con discriminante

```
-- Intensión ------------------------------------------------------------------------------------------
Vehículos = { vehículoId, marca, modelo, año, clase, numeroPuertas, tipoTransmisión, cilindrada, tipoManillar }
  PK(vehículoId)
  -- clase puede ser 'C', 'M' o 'V'
-- Extensión ------------------------------------------------------------------------------------------
Vehículos = {
  ( v1, 'Ford'     , 'F-150'    , 2021, 'V', null, null    , null,      null),
  ( v2, 'Toyota'   , 'Corolla'  , 2022, 'C',    4, 'AUTO'  , null,      null),
  ( v3, 'Honda'    , 'CBR'      , 2023, 'M', null, null    ,  600,    'SPORT'),
  ( v4, 'BMW'      , '320i'     , 2023, 'C',    4, 'Manual', null,      null),
  ( v5, 'Yamaha'   , 'R1'       , 2022, 'M', null, null    , 1000,    'SPORT'),
  ( v6, 'Chevrolet', 'Silverado', 2020, 'V', null, null    , null,      null),
  ( v7, 'Audi'     , 'A4'       , 2023, 'C',    2, 'AUTO'  , null,      null),
  ( v8, 'Kawasaki' , 'Ninja'    , 2021, 'M', null, null    ,  650, 'Cruiser'),
  ( v9, 'Mercedes' , 'Sprinter' , 2022, 'V', null, null    , null,      null),
  (v10, 'Honda'    , 'Civic'    , 2023, 'C',    4, 'CVT'   , null,      null),
  (v11, 'Ducati'   , 'Panigale' , 2023, 'M', null, null    , 1200,    'SPORT'),
  (v12, 'Nissan'   , 'Titan'    , 2021, 'V', null, null    , null,      null)
}

-- El Ford F-150, Chevrolet Silverado y Nissan Titan son pick-up (ni coche, ni moto)
-- La Mercedes Sprinter es una furgoneta (ni coche, ni moto)
```

Clasificaciones: Una relación con booleanos



Clasificaciones: Una relación con booleanos

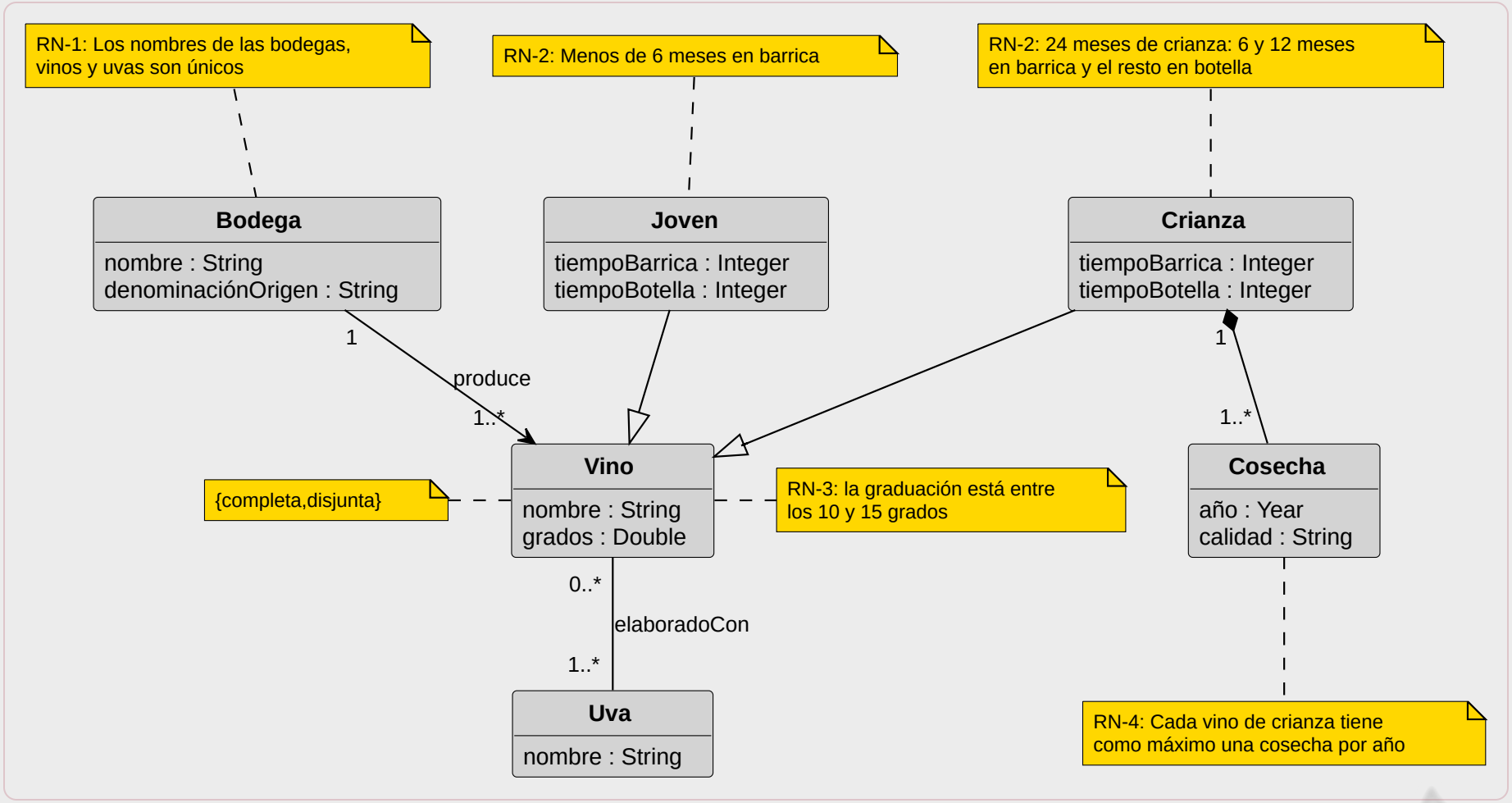
```
-- Intensión ------------------------------------------------------------------------------------------------------
RecursosHumanos= { recursoId, nombre, email, fechaContratacion, esGerente, tamañoEquipo,
  presupuesto, esIngeniero, especialidad, añosExperiencia }
  PK(recursoId)
-- Extensión ------------------------------------------------------------------------------------------------------
RecursosHumanos = {
    ( r1, 'Carlos Ruiz'   , 'carlos@empresa.com' , 2020-01-15,  true,   15, 500000.0, false, null          , null),
    ( r2, 'Laura Gómez'   , 'laura@empresa.com'  , 2021-03-10, false, null,     null,  true, 'Backend'     ,    5),
    ( r3, 'Miguel Torres' , 'miguel@empresa.com' , 2019-06-01,  true,    8, 200000.0,  true, 'Arquitectura',   10),
    ( r4, 'Patricia López', 'patri@empresa.com'  , 2022-09-15, false, null,     null, false, null          , null),
    ( r5, 'Ana Martín'    , 'ana@empresa.com'    , 2020-05-20,  true,   12, 350000.0, false, null          , null),
    ( r6, 'David Chen'    , 'david@empresa.com'  , 2021-11-08, false, null,     null,  true, 'Frontend'    ,    3),
    ( r7, 'Sofía Herrera' , 'sofia@empresa.com'  , 2018-02-14,  true,   20, 800000.0,  true, 'DevOps'      ,   12),
    ( r8, 'Roberto Vega'  , 'roberto@empresa.com', 2023-01-10, false, null,     null,  true, 'Mobile'      ,    2),
    ( r9, 'Carmen Díaz'   , 'carmen@empresa.com' , 2022-07-03, false, null,     null, false, null          , null),
    (r10, 'Luis Moreno'   , 'luis@empresa.com'   , 2019-09-25,  true,    6, 180000.0,  true, 'Security'    ,    8)
}
-- Carlos es Gerente pero no Ingeniero.
-- Miguel es Ingeniero y Gerente
-- Patricia no es ni Ingeniero ni Gerente
```

Comparación de estrategias

Estrategia	Incompleta	Completa	Disjunta	Solapada
Una tabla por clase	++	+ Muchos joins para recuperar objetos.	+ Muchos joins para recuperar objetos.	+ Muchos joins para recuperar objetos.
Una tabla por subclase concreta	-- No lo permite.	++	++	- Redundancia para atributos de objetos que pertenecen a varias subclases.
Una sola tabla (discriminante)*	+ El discriminante podría ser `null`. Muchos nulos.	+ Muchos nulos.	+ Muchos nulos.	+ El discriminante debería poder expresar pertenencia a más de una clase.
Una sola tabla (booleanos)*	+ Todos los indicadores podrían ser `false`. Muchos nulos.	+ Muchos nulos.	+ Muchos nulos.	++

Contenido

Ejemplo: Bodegas



Ejemplo: Bodegas

```
-- Una única tabla
-- Intensión
Bodegas = { bodegaId, nombre, denominacionOrigen }
  PK(bodegaId)
  AK(nombre)  -- RN-1
Vinos = { vinoId, bodegaId, nombre, grados, clase,
    tiempoBarrica, tiempoBotella }
  PK(vinoId)
  FK(bodegaId)/Bodegas
  AK(nombre)  -- RN-1
Uvas = { uvaId, nombre }
  PK(uvaId)
  AK(nombre)  -- RN-1
VinosUvas = { vinoUvaId, vinoId, uvaId }
  PK(vinoUvaId)
  FK(vinoId) / Vinos
  FK(uvaId) / Uvas
  AK(vinoId, uvaId)  -- RN-4
Añadas = { añadaId,  vinoId, año, calidad }
  PK(añadaId)
  FK(vinoId) / Vinos
  AK(vinoId, año)
```

```
-- Una única tabla
-- Extensión
Bodegas = {
  ('B1', 'Bodega Sur'  , 'Rioja'),
  ('B2', 'Bodega Norte', 'Ribera del Duero')
}
Vinos = {
  ('V1', 'B1', 'Luz Joven'  , 12.5, 'Joven'  ,  4,  8),
  ('V2', 'B2', 'Brisa Joven', 11.8, 'Joven'  ,  3,  7),
  ('V3', 'B1', 'Roble Alto' , 13.2, 'Crianza', 12, 12),
  ('V4', 'B2', 'Senda 24'   , 12.9, 'Crianza',  6, 18)
}
Uvas = {
  ('U1', 'Tempranillo'),
  ('U2', 'Garnacha'),
  ('U3', 'Cabernet Sauvignon')
}
VinosUvas = {
  ('VU1', 'V1', 'U1'),
  ('VU2', 'V1', 'U2'),
  ('VU3', 'V3', 'U1'),
  ('VU4', 'V4', 'U3')
}
Añadas = {
  ('A1', 'V3', 2021, 'Excelente'),
  ('A2', 'V4', 2022, 'Muy buena')
}
```

Ejemplo: Bodegas

```
-- Una tabla por subclase concreta
-- Intensión
Bodegas = {bodegaId, nombre, denominacionOrigen}
    PK(bodegaId)
    AK(nombre)
Uvas = {uvaId, nombre}
    PK(uvaId)
    AK(nombre)
Jovenes = {vinoId, bodegaId, nombre, grados,
    tiempoBarrica, tiempoBotella}
    PK(vinoId)
    AK(nombre)
    FK(bodegaId) / Bodegas
Crianzas = {vinoId, bodegaId, nombre, grados,
    tiempoBarrica, tiempoBotella}
    PK(vinoId)
    AK(nombre)
    FK(bodegaId) / Bodegas
VinosUvas = {vinoUvaId, vinoId, tipoVino, uvaId}
    PK(vinoUvaId)
    FK(uvaId) / Uvas
    AK(vinoId, tipoVino, uvaId)
Añadas = {añadaId, vinoId, año, calidad}
    PK(añadaId)
    FK(vinoId) / Crianzas
    AK(vinoId, año)
```

```
-- Una tabla por subclase concreta
-- Extensión
Bodegas = {
    ('B1', 'Bodega Sur'   , 'Rioja'),
    ('B2', 'Bodega Norte', 'Ribera del Duero') }
Uvas = {
    ('U1', 'Tempranillo'),
    ('U2', 'Garnacha'),
    ('U3', 'Cabernet Sauvignon') }
Jovenes = {
    ('VJ1', 'B1', 'Luz Joven'   , 12.5, 4, 8),
    ('VJ2', 'B2', 'Brisa Joven', 11.8, 3, 7) }
Crianzas = {
    ('VC1', 'B1', 'Roble Alto', 13.2, 12, 12),
    ('VC2', 'B2', 'Senda 24' , 12.9,  6, 18) }
VinosUvas = {
    ('VU1', 'VJ1', 'J', 'U1'),
    ('VU2', 'VJ1', 'J', 'U2'),
    ('VU3', 'VC1', 'C', 'U1'),
    ('VU4', 'VC2', 'C', 'U3') }
Añadas = {
    ('C1', 'VC1', 2021, 'Excelente'),
    ('C2', 'VC2', 2022, 'Muy buena') }
```

Conclusiones

La transformación de **MC a MR** traduce conceptos del dominio a esquemas relacionales **implementables**.

Cada **entidad** se convierte en una **relación**; sus atributos pasan a columnas y se define su **PK**.

Las **claves semánticas** se representan como **AK**, aunque se use un identificador artificial como PK.

El objetivo es preservar la **semántica** del modelo conceptual en el modelo relacional resultante.

Conclusiones

Las asociaciones **1:N** se implementan con **FK** en el lado N.

Las asociaciones **1:1** se implementan con una **FK** en una de las relaciones y restricciones de unicidad.

Las asociaciones **M:N** requieren una relación intermedia con dos **FK** y, normalmente, una **AK** compuesta.

Si hay rol **ordenado**, se añade un atributo de **orden** en la relación correspondiente.

Conclusiones

En clasificaciones, no existe una estrategia universalmente mejor: depende de **completitud**, **disyunción** y uso esperado.

`Una relación por clase` reduce nulos pero puede aumentar **joins**.

`Una relación única` simplifica consultas, pero suele introducir más **nulos** y un control extra (**discriminante** o **booleanos**).

El enfoque **MDD** permite automatizar parte del proceso, pero comprender las reglas manuales es clave para diseñar bien.

Gracias

Universidad de Sevilla

Departamento de Lenguajes y Sistemas Informáticos